Cascades Computer Innovation, LLC v. Sony-Ericsson Mobile Communications (USA) Inc. Doc. 77

IN THE UNITED STATES DISTRICT COURT
FOR THE NORTHERN DISTRICT OF ILLINOIS
EASTERN DIVISION

CASCADES COMPUTER INNOVATION, LLC,	)	
Plaintiff,	)	
vs.	)	**No. 11 C 7264**
DELL INC.,	)	
Defendant.	)	
CASCADES COMPUTER INNOVATION, LLC,	)	
Plaintiff,	)	
vs.	)	**No. 11 C 4574**
MOTOROLA MOBILITY HOLDINGS, INC., and SAMSUNG ELECTRONICS CO., LTD.	)	
Defendants.	)	
CASCADES COMPUTER INNOVATION, LLC,	)	
Plaintiff,	)	
vs.	)	**No. 11 C 6235**
HTC CORPORATION and LG ELECTRONICS, INC.,	)	
Defendants.	)	

Dockets.Justia.com

CASCADES COMPUTER INNOVATION, LLC,	)	
Plaintiff,	)	
vs.	)	**No. 11 C 7223**
SONY ERICSSON MOBILE COMMUNICATIONS (USA) INC.,	)	
Defendant.	)	
CASCADES COMPUTER INNOVATION, LLC,	)	
Plaintiff,	)	
vs.	)	**No. 11 C 7249**
PANTECH WIRELESS, INC.,	)	
Defendant.	)	

MEMORANDUM OPINION AND ORDER

MATTHEW F. KENNELLY, District Judge:

In separate lawsuits, Cascades Computer Innovation, LLC, has sued several defendants for patent infringement, contending that the defendants manufacture products that infringe U.S. patent number 7,065,750 (the '750 patent). Each of these claims was consolidated before the undersigned judge pursuant to Northern District of Illinois Internal Operating Procedure 13 for purposes of claim construction and, via a subsequent order, for consideration of parallel motions to dismiss filed in each of the cases. The Court previously denied the motions to dismiss, *see Cascades Computer Innovation, Inc. v. Motorola Mobility Holdings, Inc.*, Nos. 11 C 4574, 11 C 7223, 11 C

7249, 11 C 7252 & 11 C 7264, 2013 WL 3366276 (N.D. Ill. July 5, 2013), and now rules on the construction of the disputed terms in the '750 patent.

Background

The '750 patent is entitled "Method and Apparatus for Preserving Precise Exceptions in Binary Translated Code." *See* Defs.' Ex. 1 (Patent). The issue date of the patent was June 20, 2006, and the inventors listed are four individuals from Russia, who first filed the application for the patent in 1999. Cascades, the plaintiff in this case, is the exclusive licensee under the patent.

In general terms, the patent describes a method for efficiently executing on one system architecture computer programming code that is intended for a different architecture. The introductory paragraph of the "Background of the Invention" section of the '750 patent's specification describes the invention this way: "The present invention relates to a computer system executing foreign code and more particularly to a computer system and method for efficient handling of exceptions that arise when executing binary translated code." Patent at 1:36–39. The patent contains eighteen claims, two of which are at issue. Claim 1 describes the invention's binary translation system, listing six elements; Claim 15 describes a recomputing method with three elements. The parties dispute the definitions of eight terms within these two claims. For seven of the eight terms, Cascades argues that the Court need not make any construction, because the plain and ordinary meaning would be apparent to "one of ordinary skill" in the software programming field. The defendants, on the other hand, offer language from the specification for most of their proposed definitions, often arguing that the patentees sought to expressly define these terms.

Discussion

1. "Foreign code"

The term "foreign code" appears in both Claims 1 and 15 of the '750 patent. Claim 1 refers to "a non-optimizing foreign code execution module configured to maintain dedicated foreign state for each foreign binary operation executed allowing for the exceptions arisen to be handled precisely." Patent at 16:6–9. The term appears again near the end of Claim 1, where the patent states that the invention's recovery mechanism is configured "to continue foreign codes [sic] execution in case of the exception arisen during the execution of the corresponding optimized host codes." *Id.* at 16:32–35. The term also appears three times in Claim 15, each dealing with "translation of the foreign code." *Id.* at 17:23–24, 26–27, 29–30. The specification provides an express definition for the term "foreign binary code," when it states that, "[a]s used herein, foreign binary code means computer instructions written for execution on a foreign processing system but ported to the host computer system 100." *Id.* at 6:49–52.

The defendants' proposed construction of the term foreign code is: "computer instructions written for execution on a foreign processing system but ported to the host computer system." *See* Joint Notice of Modified Proposed Claim Constructions at 3 [docket no. 123]. They argue that the specification "expressly defined" the term "foreign code" to include this entire phrase. Hrng. Trans. at 39–40. Cascades argues in favor of the "plain and ordinary meaning" of this term as "understandable by one of ordinary skill in the art." *See* Joint Notice of Modified Proposed Claim Constructions at 3 [docket no. 123]. If, however, the Court determines to construe the term, Cascades proposes the

following definition: "computer instructions written for execution on a foreign processing system." *Id.* This is the same as the defendants' construction minus the last seven words, which Cascades argues are "just not necessary." Hrng. Trans. at 41. Plaintiff contends that it is obvious the patent requires porting between systems and that "porting is not part of the code," and also that the phrase at issue "is a pretty common term" among computer programmers. *Id.* at 41–43. The Court notes that at the claim construction hearing, attorneys for both sides also agreed that the Court's choice of one of these interpretations over the other will make no difference in the ultimate resolution of this dispute.[1] *See id.* at 42–43.

The Court agrees with Cascades that construction of this term is not necessary and that its meaning would be readily apparent to a computer programmer of ordinary skill. It does not take the expertise of a software engineer to determine that "code" is a bedrock concept in the field of computer programming. As the Federal Circuit has repeatedly stressed, "[c]laim terms generally are construed in accordance with the ordinary and customary meaning they would have to one of ordinary skill in the art in light of the specification and the prosecution history." *SanDisk Corp. v. Kingston Tech. Co.*, 695 F.3d 1348, 1353 (Fed. Cir. 2012). The Court does not have trouble concluding that a typical computer programmer reading the specification would understand what the claims mean by "code," and that the references to "foreign code" clearly denote code written for a system other than the host system. The claims discuss "foreign code"

[1] The parties had disagreed in their briefs over whether defendants draw their definition from the specification's definition of the correct term; Cascades contended that "foreign binary code" is not the same thing as "foreign code." But at the hearing, Cascades adopted an interpretation that uses much the same language as the defendant's construction.

and "foreign operations" in conjunction with "host operations" and "corresponding optimized host codes." *See, e.g.*, Patent at 16:10–14; 16:32–34. These terms provide the context, if it were not already apparent to a programmer, that the "foreign code" discussed in the claims is code meant for another system; further, as Cascades pointed out at the claim construction hearing, the patent's whole reason for being is to "port[ ] [code] to the host computer system." *See id.* at 6:49–52.

It is true that the patent says "foreign binary code *means* computer instructions written for execution on a foreign processing system but ported to the host computer system 100." *Id*. It is also true that the Federal Circuit has stated that "the inventor's lexicography governs" in cases where "the specification . . . reveal[s] a special definition given to a claim term by the patentee." *Phillips v. AWH Corp.*, 415 F.3d 1303, 1316 (Fed. Cir. 2005) (en banc). Such a definition, however, governs only when the patentee's definition for the term "differs from the meaning it would otherwise possess." *Id.* Neither party has argued that the long definition given to "foreign binary code" in the patent is different from what an average software programmer would have understood. As the Federal Circuit stated in *Phillips*, "[t]he inquiry into how a person of ordinary skill in the art understands a claim term provides an objective baseline from which to begin claim interpretation." *Id*. at 1313. In the case of the term "foreign code" in the context of software engineering, the Court determines there is no need to go beyond that baseline and therefore declines to construe this term.

2. "Binary translation/translator"

Claim 1 of the patent describes a "binary translation system" including "an optimizing binary translator configured to translate foreign binary operations into

optimized sequences of host operations." Patent at 16:5, 16:10–12. The term "binary translation" appears again in Claim 15, which describes a "method of recomputing a dedicated foreign state in a binary translation system from documentation generated by an optimizing translator in a case of an exception arising during execution of optimized binary translated code translated from a foreign code." *Id.* at 17:20–24. Claim 15 makes two references to "optimized binary translated code" in which the invention renders recovery points. *Id.* at 17:25–28, 31–35.

This is the sole disputed term for which Cascades does not initially advocate for the Court's adoption of its purported plain and ordinary meaning. Instead, Cascades contends that the correct definition of "binary translation" is "foreign code is processed by host software to produce new host code corresponding to the foreign code." Pl.'s Resp. at 7. Cascades argues this is "an explicit definition" from the patent itself. *Id*. It points to a passage from the specification that says "[b]inary translation means that a foreign code is processed by host software to produce new host code corresponding to the foreign code." Patent at 4:16–18. At the claim construction hearing, the defendants agreed that the Cascades construction "does come right out of the specification," but they argued that the phrase "corresponding to" in that definition adds unhelpful ambiguity to the term and "reads out" the translation element of the term. Hrng. Trans. at 44–45. Therefore, the defendants contend, their proposed definition goes further in that it helps explain what "corresponding to" means in the context of "binary translation." They propose this interpretation: "generation/generator of a new sequence of host code that performs the same functions and achieves the same behavior as on the foreign platform." Defs.' Br. at 6–7. This language, too, is drawn from the specification, in a

section that defendants say serves to "refine[ ]" the "general notion of 'binary translation' by expressly defining the term." *Id.* at 7. Cascades countered at the hearing that the defendants' interpretation does not actually define the term and that a typical software engineer would know what "corresponding to the foreign code" means.

In construing the terms of a claim, courts must "first look to, and primarily rely on, the intrinsic evidence, including the claims themselves, the specification, and the prosecution history of the patent, which is usually dispositive." *Sunovion Pharms., Inc. v. Teva Pharms. USA, Inc.*, 731 F.3d 1271, 1276 (Fed. Cir. 2013). Courts are "generally" to give the words of a claim "their ordinary and customary meaning." *Phillips*, 415 F.3d at 1312–13. However, an "exception[ ]" to this rule applies "when a patentee sets out a definition and acts as his own lexicographer." *Thorner v. Sony Computer Ent. Am. LLC*, 669 F.3d 1362, 1365 (Fed. Cir. 2012) (citing *Vitronics Corp. v. Conceptronic, Inc.*, 90 F.3d 1576, 1580 (Fed. Cir. 1996)); *see also Vitronics*, 90 F.3d at 1582 ("The specification acts as a dictionary when it expressly defines terms used in the claims or when it defines terms by implication.").

The defendants face two difficulties in advocating for their proposed definition of this term. First, the patent directly provides a meaning for "binary translation" in one of the few explicit definitions in the entire document. The patent specifically states that "[b]inary translation *means* that a foreign code is processed by host software to produce new host code corresponding to the foreign code." Patent at 4:16–18 (emphasis added). The Court has little trouble concluding that this is precisely what *Vitronics* is referring to when it says that "the specification acts as a dictionary when it expressly defines terms used in the claims." *Vitronics*, 90 F.3d at 1582. The defendants are

attempting to add to this patent-provided definition with their construction of the term, but *Phillips*, *Vitronics*, and similar cases do not suggest that a party should be permitted to add to an express definition provided in the patent. The defendants do not cite cases indicating that this is appropriate. Second, the defendants' definition comes from a passage of the specification stating what binary translation *may achieve*, not what it *is.* Here is that passage in full: "Yet another technique is to use binary translation to generate a sequence of instructions that perform the same functions and achieve the same behavior as on the foreign platform." Patent at 2:1–4. Describing a set of results from binary translation is different from stating what the term "means."

Given the direct definition of "binary translation" found in the specification of the '750 patent, the Court construes the disputed claim language as advocated by Cascades: "foreign code is processed by host software to produce new host code corresponding to the foreign code." Pl.'s Resp. at 7.

3. "Optimized/optimizing"

The term "optimize" or "optimizing" appears, in one form or another, a dozen times in the two claims at issue. In Claim 1, the described binary translation system includes both "a non-optimizing foreign code execution module" and "an optimizing binary translator configured to translate foreign binary operations into optimized sequences of host operations." Patent at 16:6, 10–12. The same claim describes a "documentation generator configured to generate a set of documentations for optimized sequences of host operations," and it also refers to a recovery mechanism that is supposed to "continue foreign codes execution in case of the exception arisen during the execution of the corresponding optimized host codes." *Id.* at 16:15–17, 26–34. The

references to the term in Claim 15 are similar, in describing an "optimizing translator" or "optimized translation," as well as "optimized binary translated code." *Id.* at 17:22–24.

The defendants propose that to "optimize" means to "extract the inherent parallelism of the foreign code." Defs.' Br. at 8. They draw this definition from the specification. *See, e.g.*, Patent at 4:24–25 ("Optimization requires extracting the inherent parallelism of the foreign code."); *id.* at 10:61–64 ("The optimizing process extracts the parallelism inherent to foreign code . . . ."). Also, the reference to two types of translators in the patent supports this construction, the defendants say. Though the optimizing translator has a parallel approach, they argue, the non-optimizing translator "eliminates the optimization by re-executing the code in a *sequential* manner" (as opposed to parallel). Defs.' Br. at 10 (citing Patent at 15:55–57). At bottom, the defendants contend, "optimization" in the '750 patent means "that the code be made to run in parallel." Defs.' Repl. at 4. At the claim construction hearing, the defendants argued that "the only optimization in this patent is [this] parallelism." Hrng. Trans. at 57. "[W]henever [the patent] refers to optimization," the defendants argued, "what it's really talking about is when it's doing that binary translation, it's putting that code . . . into parallel." *Id.* at 59. They contended that although the patent does not expressly define "optimize" in the way they propose, it does so by implication, citing *Honeywell Int'l, Inc. v. ITT Indus., Inc.*, 452 F.3d 1312, 1318 (Fed. Cir. 2006), and *C.R. Bard, Inc. v. United States Surgical Corp.*, 388 F.3d 858, 864 (Fed. Cir. 2004).

Cascades argues that the term "optimize" need not be defined for purposes of claim construction. It argued at the hearing that the word is "a well-understood term to a lay jury that is also consistent with what one of ordinary skill in the art would also

understand it to mean." Hrng. Trans. at 67–68. Cascades argues for the ordinary meaning of the term, or a generalized definition, because the word "has no particular or specialized meaning within the relevant art and is well within the understanding of a normal jury." Pl.'s Resp. at 9. To optimize, Cascades argues, is to "improve performance to run efficiently." *Id.* It cites two definitions of optimization from the American Heritage dictionary: "[t]o make as perfect or effective as possible," and "[t]o increase the computing speed and efficiency of (a program), as by rewriting instructions." Pl.'s Ex. A. It also cites a definition of the term "optimizing compiler" from Webster's 1997 *Dictionary of Computer Terms*. *See* Pl.'s Ex. B (a machine "that translates source code into machine language optimized to run as efficiently as possible on a particular microprocessor" (alterations omitted)). This, Cascades says, indicates that the definition of optimize is the same among laypeople and those of ordinary skill in computer science. Cascades proceeds to point to several passages in the patent specification where optimization is mentioned in conjunction with improving performance, and it points out that the specification uses "optimizing" to describe other processes as well. *See, e.g.*, Patent at 3:59–62 ("The present invention includes a means for optimizing the execution of binary translated code by reordering of the execution order of pending operations including memory access operations."). Cascades adds that defendants' proposed construction—"extract the inherent parallelism of the foreign code"—is a phrase that could confuse a jury.

As the Federal Circuit has observed, "[i]n some cases, the ordinary meaning of claim language as understood by a person of skill in the art may be readily apparent even to lay judges, and claim construction in such cases involves little more than the

application of the widely accepted meaning of commonly understood words." *Phillips*, 415 F.3d at 1314. Such terms "are generally given their ordinary and customary meaning" according to "a person of ordinary skill in the art in question at the time of the invention." *Id.* at 1312–13. However, "[i]diosyncratic language, highly technical terms, or terms coined by the inventor are best understood by reference to the specification." *3M Innovative Props. Co. v. Tredegar Corp.*, 725 F.3d 1315, 1321 (Fed. Cir. 2013). In addition, a patentee need not expressly define a term in order for a definition of that term to be drawn from the patent. Instead, a definition "may be inferred from clear limiting descriptions of the invention in the specification or prosecution history." *Aventis Pharma S.A. v. Hospira, Inc.*, 675 F.3d 1324, 1330 (Fed. Cir. 2012). Yet courts must take care not to infer that a patentee intended to define a term merely because only a single example or embodiment of the term is described in the patent itself. The Federal Circuit "has expressly rejected the contention that if a patent describes only a single embodiment, the claims of the patent must be construed as being limited to that embodiment." *Liebel-Flarsheim Co. v. Medrad, Inc.*, 358 F.3d 898, 906 (Fed. Cir. 2004) (collecting cases). The Federal Circuit recently had the occasion to again address how courts are to navigate this area of claim construction:

> It is . . . not enough that the only embodiments, or all of the embodiments, contain a particular limitation. We do not read limitations from the specification into claims; we do not redefine words. Only the patentee can do that. To constitute disclaimer, there must be a clear and unmistakable disclaimer.

Thorner, 669 F.3d at 1366–67.

The cases the defendants cite stand for the proposition that a definition can, in certain circumstances, be taken from a specification even if it does not expressly define

the particular term. In *Honeywell*, the Federal Circuit held that limitation of the term "fuel injection system component" to mean a fuel filter (and nothing else) was appropriate. *Honeywell*, 452 F.3d at 1318. The court concluded that the concept of a fuel filter "was not merely discussed as a preferred embodiment" in the specification because "the written description refer[red] to the fuel filter as 'this invention' or 'the present invention'" at least four times. *Id.* (citing such language as, "This invention relates to a fuel filter for use in the fuel line that delivers fuel to a motor vehicle engine."). This contextual evidence helped convince the court that the fuel filter was the only embodiment of the claim term in the invention. Only one other fuel component was mentioned in the description (a fuel line), but that component "was not required by the patentee to be made of an electrically conductive polymer material, as the claims require." *Id.* Given these facts, it was proper "to take the patentee at his word and the word was that the invention is a fuel filter." *Id.*

In *C.R. Bard*, the term at issue referred to a medical device that included a plug. *C.R. Bard*, 388 F.3d at 860. The "Summary of the Invention" section of the patent specification stated that the invention was "an implantable prosthesis," then went on to say that "[t]he implant includes a pleated surface." *Id.* Similarly, the patent's abstract referred to "[a]n implantable prosthesis including a conical mesh plug having a pleated surface." *Id.* at 860–61. The court concluded that although the patent did not expressly define the plug, these statements "unequivocally define[d] the claimed plug as having pleats" in two places where it "describe[d] in general terms what it deem[ed] to be the invention." *Id.* at 864. The Federal Circuit found the placement of these statements helpful in determining their significance. Although placement of a description in that

section of a patent is not "determinative," it "can signal the likelihood that the statement will support a limiting definition of a claim term." *Id.* However, the court cautioned that the weight given to such language "must, of course, be determined on a case-by-case basis." *Id.* In *C.R. Bard*, there were "[s]tatements of general applicability" that "clearly define[d]" the plug as having pleats, which was the end of the matter. *Id.* at 866.

In the present case, the difficulty with the defendants' interpretation is that they draw it from passages of the patent describing the end result of optimization, and not what optimization, as a process, actually involves. There is no direct description of the process of optimization, as there was for the clear definition of the plug in *C.R. Bard* or the concrete characterization in *Honeywell*. For example, the first passage the defendants label as definitional in their brief ("[o]ptimization requires extracting the inherent parallelism of the foreign code," Patent at 4:24–25) does not actually say what optimization is or what it does; rather, the passage merely says what optimization requires. A statement of what a process requires is not the same as defining that process; for example, doing homework might require putting pen to paper, but identifying that function does not provide a definition of what homework is. The same is true of the other passages the defendants cite, *e.g.*, "[t]o exploit the explicit parallelism of this architecture . . . it is necessary to optimize the binary translated code in a manner that maintains precise exceptions." *Id.* at 8:61–64. And indeed, one passage of the specification makes clear that optimization and extracting parallelism are two different things, as one can produce the other: "[T]o exploit parallelism of the host processor architecture in binary translated code, the host code must be optimized." *Id.* at 4:22–24.

It therefore seems fairly clear that the patentees did not seek to expressly define the term "optimize" and that the passages defendants cite cannot be used for that purpose. Though the term is mentioned multiple times in conjunction with extracting or exploiting the parallelism of a particular system architecture, it does not follow that such extraction or exploitation is the only means of optimizing, nor that the patentee intended it to be. The defendants may be right that running translated code in parallel is the *sine qua non* of this particular patent, see Defs.' Repl. at 5, but that does not mean that "optimize" has the specific definition they wish to assign it. As Cascades points out, "optimize" is not always mentioned in direct conjunction with exploiting parallelism, but rather with improvement of performance. In one of the embodiments, the "optimizing binary translation process" is described as one intended "to improve performance"; no mention is made of code running in sequence or parallel. Patent at 7:12–15; *see also id.* at 7:56–59. Another passage states that *performance* can be optimized, which further argues against adopting the defendants' specific definition: "to optimize performance, load operations can be moved ahead of store operations." *Id.* at 7:22–23.

The Court concludes that the defendants have not offered sufficiently compelling evidence that the patentees established a "lexicography" encompassing this term that differs from its plain meaning as understood in the computer programming field. The Court concludes that the term "optimizing" in the '750 patent, though used frequently in association with extracting parallelism, is not defined by that function, nor is it "[i]diosyncratic," "highly technical," or "coined by the inventor," *3M Innovative Props.*, 725 F.3d at 1321. Rather, the term requires no construction beyond its ordinary and customary meaning to a person in the art at the filing date of the patent, *see Phillips*,

415 F.3d at 1312–13—that is, to make the most effective use of, or to make most efficient.

4. "Documentation"

Apart from the disputed phrase "documentation generator configured to generate," discussed below, the term "documentation" (or a variant) appears fifteen times in the claims at issue. It is used as both a singular and plural noun, sometimes in the same sentence. For example, Claim 15 describes a recomputing method that "generat[es] a set of documentations during the optimized translation of the foreign code, wherein each documentation in the set of documentations corresponds to a recovery point in the optimized binary translated code . . . ." Patent at 17:29–32. Claim 1 refers to "documentation" as an indefinite noun: "for each host operation address, operations required to calculate a corresponding foreign state for the host operation address are added to documentation." *Id.* at 16:22–25. Claim 15 also describes part of the invention as a "method of recomputing a dedicated foreign state in a binary translation system from documentation generated by an optimizing translator." *Id.* at 17:20–22. "Documentation" also serves as a modifier; in Claim 1, there is reference to "a documentation tracker configured to record host operation addresses at appointed points of the host operation sequences being executed," *id.* at 16:20–22, a function that differentiates the tracker from the "documentation generator."

The specification states that "[e]very recovery point is described by a documentation set . . . that contains information where all foreign registers are located in the host registers in the optimized binary translated code." Patent at 9:35–41. The defendants draw from this passage nearly verbatim for their proposed definition of the

term documentation: "data describing where all foreign registers are located in the host registers in the optimized binary translated code." Joint Notice of Modified Proposed Claim Constructions at 2 (docket no. 123). They cite similar language in Claim 1, *see* Patent at 16:15–19, and Claim 15, *see id.* at 17:30–34. At the claim construction hearing, the defendants argued that the definition of "documentation" cannot be "generic[ ]," because a generic definition of the term "may not be enough information within the context of the patent to recreate that foreign state which is essential to move on in the program." Hrng. Trans. at 71. The defendants contended that it is essential for the definition to reflect that a single documentation contains information about *all* registers related to a single recovery point. Furthermore, the defendants argued the patent actually imposes two requirements on a documentation: that it include "a list of registers with values in them," but also "operations to calculate the foreign state using those values." *Id.* at 81.

At the claim construction hearing, Cascades contended that this term requires no construction. That is because "the claim itself tells you what documentation is," Cascades argued, pointing to passages from both Claims 1 and 15. *Id.* at 77. The passage Cascades cites from Claim 1 states that "each documentation describes operations required to calculate a corresponding foreign state for an appointed point." Patent at 16:17–19. The cited passage from Claim 15 says that "each documentation . . . corresponds to a recovery point in the optimized binary translated code and describes operations required to calculate a corresponding foreign state for the recovery point." *Id.* at 17:30–34. Cascades argues that defendants' proposal is actually a definition for a documentation *set*, not a *documentation*. It is incorrect,

Cascades argues, to define the term as data describing the location of *all* foreign registers, because each documentation "only requires calculating a foreign state for a given, 'appointed point' (in the case of claim 1)." Pl.'s Resp. at 12. (Cascades did not, however, explain how a "documentation set" is different from a "documentation.")

Defendants reply that Cascades conflates the plural "set of documentations" with the singular "documentation set" or "set of documentation." "When discussing a plurality of documentations, the '750 patent consistently calls that plurality a 'set of documentations.'" Defs.' Repl. at 8. Elsewhere, the patent uses both "documentation" and "documentation set" "to describe a single documentation associated with a single Recovery Point that describes the location of the foreign registers associated with that Recovery Point." *Id.* at 7. Defendants also argue that the plain and ordinary meaning of "documentation" is inappropriate for purposes of constructing the claims at issue, because the patent "describes a very specific and proprietary 'documentation.'" *Id.* at 9.

Neither party's proposal appears to capture the meaning of the term, which as described above is used in different ways in the patent. Defendants seem to want to narrow what would appear to be a general term to a particular contextual usage. Though the defendants draw from the specification for their definition, there are other uses of the term in the patent that are not as particularized as the passage the defendants use. But Cascades' approach, which is to say that no construction is needed, is not particularly helpful given the varying usages of the term. Cascades argued at the claim construction hearing that no interpretation is needed because the claims themselves define "documentation." As indicated, Cascades cites a passage from Claim 1 stating that "each documentation describes operations required to

calculate a corresponding foreign state for an appointed point," Patent at 16:17–19, and a passage from Claim 15 stating that "each documentation . . . corresponds to a recovery point in the optimized binary translated code and describes operations required to calculate a corresponding foreign state for the recovery point." *Id.* at 17:30–34. But neither of these passages defines documentation; rather, they describe what certain documentation does in particular situations.

The Court has returned to the patent for guidance on what the inventors meant by the term and how a person of ordinary skill in the field would have understood it. The specification contains two examples of "documentation," which help illustrate what it is. The specification states in columns 13 and 14:

> There are three Recovery Points in the code. The first one describes register contents before starting execution. The second point (second wide instruction) is described by the following documentation:

Foreign register	Host register
EDX	R1
ECX	R2
ESI	(not changed)
FLAGS	R3

> The second point is described by this documentation because registers R1, R2 and R3 have been released by the host optimizing scheduler and then reused in further calculations. After finishing execution of the code the documentation will have the following contents:

Foreign register	Host register
EDX	R1
ECX	R2
ESI	R4
FLAGS	R5

Patent at 13:50-67 & 14:1-8.

As shown by these illustrative examples, the term "documentation" has a far more generalized meaning than the very specific one that defendants seek to ascribe to it. The Court rules that documentation, as used in the present context, simply means written text containing information that describes a software operation. Further particularization of the type, nature, or contents of any given documentation is provided by the claim language or from—for example, in the fourth element of claim 1, the documentation is said to "describe[ ] operations required to calculate a corresponding foreign state for an appointed point." Patent at 16:17-19. It would be both unnecessary and inappropriate to load this sort of particularization into the definition of the term "documentation" itself.

5. "Documentation generator configured to generate"

The term "documentation generator configured to generate" appears in Claim 1 of the patent, where it is listed among the elements included in the patent's binary translation system. The claim says that the system comprises "a documentation generator configured to generate a set of documentations for optimized sequences of host operations, wherein each documentation describes operations required to calculate a corresponding foreign state for an appointed point." Patent at 16:15–19.

For this term, neither party offers a specific definition. The defendants argue that "documentation generator configured to generate" is a "means-plus-function term" within the meaning of 35 U.S.C. § 112(f). They contend that because the term itself evokes no specific structure, and because structure for the term cannot be found in the specification, the term is indefinite, and thus the claim containing it (Claim 1) is invalid. In their opening brief, the defendants implied that the patent should have included some

sort of algorithm for the documentation generator, and they expanded on this argument at the hearing, arguing that algorithms should receive "some degree of heightened scrutiny." Hrng. Trans. at 87. To support this notion, they cited *Aristocrat Techs. Australia Pty Ltd. v. Int'l Game Tech.*, 521 F.3d 1328, 1334 (Fed. Cir. 2008), a case where the claim's description of the structure for a computer-implemented function went "no farther than saying that the claimed functions are performed by a general purpose computer" and was thus invalid. There, the Federal Circuit noted that plaintiffs need not "produce a listing of source code or a highly detailed description of the algorithm to be used to achieve the claimed functions in order to satisfy 35 U.S.C. § [112(f)]." *Id.* at 1338. But they must "at least disclose the algorithm that transforms the general purpose microprocessor to a special purpose computer programmed to perform the disclosed algorithm." *Id.* (internal quotation marks omitted). At the hearing, the defendants also cited *Ex Parte Rodriguez*, No. 2008-000693, slip op. at 20–27 (B.P.A.I. Oct. 1, 2009), *available at* http://www.uspto.gov/ip/boards/bpai/decisions/prec/fd080693.pdf, a case from the Board of Patent Appeals and Interferences, in which claims containing the term "system configuration generator" were found invalid because the term was indefinite.

Cascades says this term should be given its plain and ordinary meaning as understandable by one of ordinary skill in this field. Cascades also argues that the burden to establish indefiniteness "by clear and convincing evidence" belongs to defendants, "including [when arguing] a lack of alleged structure." Pl.'s Resp. at 16 (citing *Budde v. Harley Davidson, Inc.*, 250 F.3d 1369, 1376 (Fed. Cir. 2001)). Because "[d]efendants have not provided <u>any</u> evidence" about the understanding of a "person of

skill" with regard to a "documentation generator," Cascades says, they have not met their burden to prove that the term is indefinite. *Id.* At the claim construction hearing, Cascades argued that the Court therefore does not have to consider whether the patent provides an algorithm for the "documentation generator" term in accordance with the *Aristocrat* case. Cascades also argues that the specification sufficiently describes structure for a documentation generator, an argument that the Court will address in detail below.

Under 35 U.S.C. § 112(f), formerly known as section 112, paragraph 6, "[a]n element in a claim for a combination may be expressed as a means or step for performing a specified function without the recital of structure, material, or acts in support thereof." The statute then states that "such claim shall be construed to cover the corresponding structure, material, or acts described in the specification and equivalents thereof." *Id.* The Federal Circuit has stated that the application of section 112(f) is a "straightforward" question. *Inventio AG v. ThyssenKrupp Elevator Am. Corp.*, 649 F.3d 1350, 1356 (Fed. Cir. 2011). If a claim term contains the word "means," a rebuttable presumption that section 112(f) governs construction of the term is triggered; if the word "means" is absent, a presumption is triggered that section 112(f) does not govern. *Id.* This latter presumption is "a strong one that is not readily overcome." *Lighting World, Inc. v. Birchwood Lighting, Inc.*, 382 F.3d 1354, 1358 (Fed. Cir. 2004). The Federal Circuit recently put this presumption in strong terms: "When the claim drafter has not signaled his intent to invoke § [112(f)] by using the term 'means,' we are unwilling to apply that provision without a showing that the limitation essentially is

devoid of anything that can be construed as structure." *Flo Healthcare Solutions, LLC v. Kappos*, 697 F.3d 1367, 1374 (Fed. Cir. 2012).

In evaluating whether this presumption should stand, "it is sufficient if the claim term is used in common parlance or by persons of skill in the pertinent art to designate structure, even if the term covers a broad class of structures and even if the term identifies the structures by their function." *Lighting World*, 382 F.3d at 1359–60. "Ultimately, whether claim language invokes § [112(f)] depends on how those skilled in the art would understand the structural significance of that claim language, assessed against the presumptions that flow from a drafter's choice to employ or not employ the term 'means.'" *Inventio*, 649 F.3d at 1360. Courts may "review[ ] the intrinsic record, as well as extrinsic evidence such as technical dictionaries, to determine if the challenger successfully rebutted the presumption that a claim that lacks the term 'means' is not subject to § [112(f)]." *Id.* at 1357. In terms of the burden a challenging party must meet to prove indefiniteness in the context of a means-plus-structure dispute, "[t]he party alleging that the specification fails to disclose sufficient corresponding structure must make that showing by clear and convincing evidence." *TecSec, Inc. v. IBM Corp.*, 731 F.3d 1336, 1349 (Fed. Cir. 2013).

To sum up, the law presumes that the term "documentation generator" is not subject to section 112(f) because the term does not use the word "means." To overcome the presumption, defendants' burden is to provide evidence that a typical software programmer would not understand that the term documentation generator connotes structure. The Court acknowledges that the term "documentation generator" is not directly defined in the patent, nor is it particularly revealing to the uninitiated. But

the defendants nonetheless have not produced persuasive evidence, as is their burden, that a typical software engineer would not understand what structure lies behind a documentation generator as included in the '750 patent. This is so in light of Cascades' contention that there is some structure for the generator in the patent and its allusions to knowledge in the relevant field.

In this case, the term appears only once in the entire patent, in one of the disputed claims. At no time does the specification otherwise reference a "documentation generator," although there are, as discussed above, many references to "documentation." In light of the absence of a definition for the term in the specification, Cascades contends that one is not necessary, because the average computer programmer would understand the structure of a documentation generator. In response, the defendants do not directly argue the contrary, i.e., that a typical software engineer would not understand what the structure of a documentation generator is. Instead, they cite several cases, some of which concern construction of terms including the word "generator" (though none concerns a documentation generator) and others of which concern arguably similar terms, such as "processor." These cases are not directly relevant to the matter at hand. In *Rodriguez*, an administrative decision that is not controlling authority, the Board of Patent Appeals and Interferences invalidated several claims that included the term "system configuration generator." *Ex Parte Rodriguez*, slip op. at 20–27. In that case, unlike this one, the side arguing in favor of the patent's validity had not "ever suggested that these elements are a known structure in the prior art" and in fact "argue[d] that these elements are of their invention and not known in the prior art." *Id.* at 27. Thus, although the defendants label *Rodriguez* "the

best case" on the indefiniteness of "generator" as a term connoting structure, Hrng. Trans. at 82, the facts and arguments there were different.[2] As *Flo Healthcare* and other Federal Circuit cases on the means-plus-function inquiry make clear, analysis on this question must be case-specific.

The facts were also different in *Aristocrat*, a case the defendants cite multiple times for the proposition that a claim including software must disclose a unique algorithm for that software. There, the disputed terms used the word "means," which meant the court did not have to consider the presumption that section 112(f) does not apply when that word does not appear. *Aristocrat*, 521 F.3d at 1331 (terms were "game control means" and "control means"). In fact, the parties in that case agreed that the terms at issue were means-plus-function terms. *Id.* The court therefore did not have to evaluate a crucial support for that presumption: whether "the claim term is used in common parlance or by persons of skill in the pertinent art to designate structure." *Inventio*, 649 F.3d at 1359–60. Instead, the plaintiff in *Aristocrat* argued that "devising

[2] The defendants also cite two district court cases that mention "generator" terms. In one, the court did not appear to consider whether the term at issue, "report generator for outputting," would reveal sufficient structure to one of ordinary skill in the field. Indeed, the court did not discuss the presumption against application of section 112(f) when the word "means" does not appear in the claim term. *See Isogon Corp v. Amdahl Corp.*, 47 F. Supp. 2d 436, 449–50 (S.D.N.Y. 1998). Of course, that case was decided before *Lighting World*, *Inventio AG*, *Flo Healthcare* or any of the other cases discussing the presumption cited here. The defendants cite another district court case predating those decisions, but there, the parties agreed that the term "state generator" was a means-plus-function term, and thus the court did not have to decide whether the presumption applies. *See QSIndustries, Inc. v. Mike's Train House, Inc.*, 230 F. Supp. 2d 1240, 1245–46 (D. Or. 2002). The defendants also cite a "triumvirate of recent decisions" from the Patent Trial and Appeal Board, but each of those cases construed the term "processor," and the defendants make no argument explaining why "processor" and "documentation generator" are equivalent for this purpose. *See* Defs.' Br. at 14 (citing *Ex Parte Erol*, No. 2011-001143, 2013 WL 1341107, at *9 (P.T.A.B. Mar. 11, 2013); *Ex Parte Lakkala*, No. 2011-001526, 2013 WL 1341108, at *7 (P.T.A.B. Mar. 11, 2013); *Ex Parte Smith*, No. 2012-007631, 2013 WL 1341109, at *8 (P.T.A.B. Mar. 12, 2013)).

an algorithm to perform [the invention's] function would be within the capability of one of skill in the art." *Aristocrat*, 521 F.3d at 1334.

That is not what Cascades is arguing here. Rather, it contends that "those of skill understand that a 'documentation generator' is a structure, i.e., a programming tool." Pl.'s Resp. at 15. To support this argument, Cascades offers definitions of "document" and "generator" from a scientific and technical dictionary; the dictionary notes that "generator" is a computer science term and defines it as a "program that produces specific programs as directed by input parameters." Pl.'s Ex. D at 4. Cascades also cites two Wikipedia pages: one defining "documentation generator," and another providing a list of documentation generators produced by various programmers and companies. *See* Pl.'s Exs. E, F. The defendants have attacked these sources as dated after the patent application and "'less significant' than the intrinsic evidence," Defs.' Repl. at 10. The Court agrees that these sources are not perfect, but they do provide some support for the proposition that a programmer of ordinary skill would understand the structure of a documentation generator. By contrast, the defendants have not provided evidence that programmers in the field would *not* understand the structure of a documentation generator.

For its part, Cascades also contends that there is at least one example of the structure of a documentation generator in the specification itself, where it describes a "dynamic binary translator" that "can generate host code for every foreign instruction in sequential order with the following properties. Dynamic binary translator uses a subset of host registers in register file 100 (FIG. 1) to map the foreign registers into the host registers." Pl.'s Resp. at 15–16 (quoting Patent at 8:65–9:3). At the hearing, Cascades

argued that this description "is a structure of a documentation generator." Hrng. Trans. at 94. The defendants respond that "dynamic binary translator" is already a claim term (although it is not, strictly speaking; the disputed claim term is "binary translation/translator," without the "dynamic"). Cascades is not arguing, however, that the passage it points to provides *the* structure of a documentation generator; rather, it is "an example of a generic structure," which "extracts information from the host code and maps foreign registers into the host registers for creating documentation." Pl.'s Resp. at 15. In addition, elsewhere in the specification, the patentees wrote that "[t]he documentation is created during the optimizing binary translation processes 202," pointing to one of the diagrams in the patent. Patent at 15:17–18.

The Court cannot hold the "documentation generator" term invalid for indefiniteness without evidence from the defendants that computer programmers would not understand the structure of Claim 1's documentation generator in the context of the whole patent. This conclusion is supported by the Federal Circuit's admonition that the presumption against construing a claim term without "means" as means-plus-function term "is a strong one that is not readily overcome." *Lighting World*, 382 F.3d at 1358. Cascades has invoked this presumption, and it has pointed to an example of a structure for a documentation generator in the patent itself. In response, the defendants have pointed to inapposite cases that do not involve the term "documentation generator" and in which the patent uses the word "means" or the parties agreed that a particular term was a means-plus-function term. For all of these reasons, the Court agrees with Cascades that this term is not subject to section 112(f), and that its definition is in line

with what Cascades says a typical computer programmer would understand a documentation generator to be: a programming tool that produces documentation.

6. "Operations required to calculate a corresponding foreign state"

The term "operations required to calculate a corresponding foreign state" appears in both Claim 1 and Claim 15. In Claim 1, it is included in the context of the "documentation tracker," which is "configured to record host operation addresses at appointed points of the host operation sequences being executed, wherein, for each host operation address, operations required to calculate a corresponding foreign state for the host operation address are added to documentation." Patent at 16:20–25. The term is similarly situated within Claim 15, although it is in reference to a "recovery point" and not "appointed points." Claim 15 describes a recomputing method that generates a "set of documentations," "wherein each documentation in the set of documentations corresponds to a recovery point in the optimized binary translated code and describes operations required to calculate a corresponding foreign state for the recovery point." *Id.* at 17:29–34.

Defendants' proposed definition includes the entirety of the term at issue, but then adds this clause: "but not by 'roll[ing] back its target registers to recover state.'" *See* Joint Notice of Modified Proposed Claim Constructions at 2 [docket no. 123]. They say this limitation comes from the patent's prosecution history, where, defendants argue, "the patentee made a definitive statement that distinguished its 'operations required to calculate a corresponding foreign state' from the prior art's way of 'roll[ing] back its target registers to recover state.'" Defs.' Br. at 16. Specifically, defendants argue that the patent examiner rejected some of the claims that did not make it into the

final patent, because they were "anticipated by" another patent, the Kelly patent. *See* Defs.' Ex. 2 at 84–87, 92–94. Cascades, as elsewhere, advocates for use of the plain and ordinary meaning of this phrase rather construing it, but alternatively proposes this definition: "the actions resulting from instructions required to determine a corresponding non-native state." *See* Joint Notice of Modified Proposed Claim Constructions at 2–3 [docket no. 123].

The Kelly patent used an operation called a "rollback" when handling exceptions, and defendants say the patent examiner initially rejected what is now Claim 1 of the '750 patent because of Kelly. At the time, the defendants argue, the '750 patent did not include the phrase at issue here ("operations required to calculate a corresponding foreign state"); they say the patentees added it after the rejection. "[O]n the basis of that amendment, [patentees] argued that the claims were novel over Kelly and Kelly's rollback feature described above." Defs.' Br. at 17. The defendants cite the applicants' statement to the Patent Office contrasting the Kelly patent based specifically on the roll-back language, which they argue is used in context of the "operations required to calculate a corresponding foreign state" term. *Id.* (citing Defs.' Ex. 2 at 75.) Defendants argue that the definition of this term therefore must include the limitation that distinguishes the '750 patent from Kelly.

At the hearing, Cascades argued that "in fact, the patentee never disclaimed the rollback operation." Hrng. Trans. at 102. They argued that "[i]nstead, the patentee said that just because Kelly has this rollback operation does not mean that documentation is inherently disclosed in Kelly." *Id.* at 104. As support, Cascades cites the same page of the letter just referenced, in which the applicants stated that "it is not inherent in Kelly to

store a set of documentations each including operations for host operation addresses . . . ." Def.'s Ex. 2 at 75. Cascades argued at the hearing that "[t]he patentee never said that if there is documentation . . . and the described operations, then there can be no rollback. He simply stated that Kelly does not disclose the documentations, therefore, it's distinguishable." Hrng. Trans. at 105. In its brief, Cascades also contends that negative limitations like the one defendants propose are disfavored by the Federal Circuit without "a firm anchor in the specific claim language of the specification of the patent." Pl.'s Resp. at 17 (citing *Linear Tech. Corp. v. ITC*, 566 F.3d 1049, 1060 (Fed. Cir. 2009)).

There is a "heavy presumption that claim terms carry their full ordinary and customary meaning, unless [the defendant] can show the patentee expressly relinquished claim scope" during the prosecution of the patent. *Epistar Corp. v. Int'l Trade Comm'n*, 566 F.3d 1321, 1334 (Fed. Cir. 2009). The alleged disavowal must "be both so clear as to show reasonable clarity and deliberateness, and so unmistakable as to be unambiguous evidence of disclaimer." *Omega Eng'g, Inc. v. Raytek Corp.*, 334 F.3d 1314, 1325 (Fed. Cir. 2003) (citation omitted). The Federal Circuit has "consistently rejected prosecution statements too vague or ambiguous to qualify as a disavowal of claim scope." *Id.* (collecting cases)). To qualify, though a "disavowal of claim scope" can be made "through arguments made to distinguish prior art references," these arguments must "constitute clear and unmistakable surrenders of subject matter." *Cordis Corp v. Medtronic Ave, Inc.*, 511 F.3d 1157, 1177 (Fed. Cir. 2008).

The disputed page of the prosecution history is part of a December 2005 submission by applicants to the patent examiner supporting what eventually became the

'750 patent. The examiner had found claim 35 (which ended up as claim 1 of the '750 patent) to be anticipated by the Kelly patent. The examiner noted that although the Kelly patent did not explicitly disclose a "documentation tracker configured to track an executable path" to be used when an exception occurs, it did disclose "code morphing software that handle exceptions and errors . . . by replacing working state with correct target state as necessary." Defs.' Ex. 2 at 87. The examiner stated that "[i]t is inherent that some type of 'documentation' is kept by the executing system to track successful / unsuccessful execution of code segments, used to evaluate and correct binary translations." *Id.*

In response, the applicants amended claim 35 in an effort to address the examiner's finding. Among other things, they added the language currently in question: "operations required to calculate a corresponding foreign state for the host operation address are added to documentation." *Id.* at 69.

Together with their amendments, the applicants offered a narrative submission in which they sought to distinguish the claim, as amended, from the Kelly patent. The applicants first described how the Kelly patent deals with exceptions: "Kelly discloses transferring the contents of all working registers to official target registers and allowing an operation called a rollback to quickly transfer the content of all official target registers back to their working register equivalents." Defs.' Ex. 2 at 75. "If an exception occurs [in the Kelly invention], <u>the original state in the target registers</u> at the last update (or commitment) may be recalled to the working registers." *Id.* The applicants then contrasted their claim: "In contrast, claim 35 'discloses adding <u>operations required to calculate</u> a corresponding foreign state for the host operation address are added [sic] to

documentation." *Id.* (emphasis in original). If an exception occurs, the operations are executed to recover a foreign state and then continue execution of the foreign code. *Id.*

The applicants contended in their submission to the examiner that Kelly was different, because it "discloses rolling back contents of official target registers back to the working registers' equivalents . . . . Kelly does not need to use documentation to perform a set of operations because Kelly rolls back its target registers to recover state." *Id.* The applicants also addressed the examiner's statement that it was "inherent" that Kelly used documentation to allow operations to continue: they stated that it was not inherent in Kelly to store a set of documentations and then to determine a documentation to recover a foreign state, again, "because Kelly rolls back its target registers to recover state." *Id.*

It is clear that the patent applicants sought to differentiate their invention from Kelly by pointing out that Kelly does not use documentation to perform recovery operations after an exception because it rolls back its target registers to "recover state." The question, however, is whether this amounts to a clear and unequivocal statement that the reference in the '750 patent to "operations required to calculate a corresponding foreign state" *excludes* rolling back the target registers. The Court concludes that the patentees did, in fact, clearly and unmistakably disclaim the use of a rollback function. They did so by their statements to the patent examiner in which they expressly differentiated their patent's methodology from the rollback function employed by the Kelly patent. In essence, the patentees said that how the Kelly patent operated in this regard and how their patent operated were two distinct ways of accomplishing a particular goal. Because the patentees differentiated the two patents in order to

distinguish a prior art reference that otherwise would have invalidated their patent, they clearly and unmistakably surrendered the use of a rollback functionality.

For these reasons, the Court adopts defendants' proposed limitation on this claim term. The "operations required to calculate a corresponding foreign state" as described in claim 1 do not include rolling back the target registers to recover an earlier state.

7. "Precisely"

The word "precisely" appears just once in the claims at issue in this case. In Claim 1 of the patent, there is description of a "binary translation system." The first element of that system is "a non-optimizing foreign code execution module configured to maintain dedicated foreign state for each foreign binary operation executed allowing for the exceptions arisen to be handled precisely." Patent at 16:6–9.

Defendants propose a detailed definition for this seemingly commonplace word: "a later instruction has not yet been executed and all prior instructions preceding the instruction causing the exception have executed and committed their results." Defs.' Br. at 18. That passage comes from a section of the specification describing an example of how the system works, dealing with hypothetical instructions "ie" and "im": "The exception generated by instruction 'ie' is a 'precise' exception if instruction 'im' has not yet been executed and all prior instructions preceding instruction 'ie' have executed and committed their results." Patent at 8:26–29. Defendants contend that the inventors acted as their own lexicographer for the term "precise" by writing this passage, because "[t]he patent takes great effort to explain exactly what 'precise' means in the context of its invention." Defs.' Repl. at 13.

As with other terms, Cascades argues that "a person having ordinary skill in the art at the time of the invention would understand 'precisely' to have its plain and ordinary meaning." Pl.'s Resp. at 21. It says the term "describe[s] how exceptions *are handled precisely*"—a concept that Cascades says is different from a "precise exception," which it says the defendants draw from for their definition. *Id.* at 19. "[A]n exception is 'handled precisely' when the behavior of foreign code, including precise exceptions, is maintained." *Id.* (This is apparently what Cascades means by "plain meaning.") Cascades then argues, without quoting the patent, that "[p]recise exceptions can be defined as exceptions when running binary translated code that are processed by the host system to correctly emulate the foreign architecture." *Id.* at 20. Cascades also contends that defendants draw their definition from a part of the patent that is discussing "speculative mode," which it argues is "not the proper definition of 'precisely.'" *Id.* at 21. At the claim construction hearing, Cascades argued that the plain meaning of "precisely" is adequate, because the term is "quite well-described in Claim 1." Hrng. Trans. at 108.

One difficulty in the use of "precisely" in Claim 1 is that the term does not appear anywhere else in the patent. However, the term "precise" appears many other times, including in the title of the patent: "Method and Apparatus for Preserving Precise Exceptions in Binary Translated Code." The specification often refers to the importance of "maintaining precise exceptions," and "[t]he ability to preserve the behavior of the foreign code, including precise exceptions." Patent at 3:31, 23–24. "[I]t is desirable to support precise exceptions in a host architecture in a manner that correctly emulates a foreign architecture." *Id.* at 3:50–52. Further, the specification describes "[t]he present

invention" as a way to execute binary translated code "in a manner . . . that supports precise exception maintenance." *Id.* at 3:56–59. Although Cascades appears to argue that a "precise exception" and an exception that is "handled precisely" are different things, there is not one instance in the patent in which "precise" is used as anything but a modifier for the word "exception." It would not appear to be a stretch to conclude that "precise exception maintenance" and handling an exception "precisely" are similar concepts, and perhaps the same concept.

For these reasons, adoption of the defendants' proposed definition appears appropriate. Though it comes from a definition of "precise" in the context of an exception, Cascades does not present any good reason why a definition for the term "precisely" cannot be drawn from the patent's use of the term "precise." The defendants' definition does come directly from the patent, specifically, from a passage where the patent declares when an exception "is" precise: "The exception generated by instruction 'ie' is a 'precise' exception if instruction 'im' has not yet been executed and all prior instructions preceding instruction 'ie' have executed and committed their results." Patent at 3:8–12. The Cascades definition, that "precisely" means the behavior of the foreign code is maintained, has a few disadvantages. Though the notion of maintaining a foreign state appears in the same clause of Claim 1, that clause notes that handling exceptions precisely is something *allowed by* maintaining a dedicated foreign state. *See id.* at 16:6–9. That is not the same thing as defining "precisely." Further, defendants' definition is more comprehensive, in that it specifically describes what must happen for an exception to be considered precise.

The Court therefore adopts the defendants' construction of "precisely," but adds the words "such that" at the beginning of the definition in order to make the interpretation grammatically correct; "precisely" is an adverb. That construction is: "such that a later instruction has not yet been executed and all prior instructions preceding the instruction causing the exception have executed and committed their results."

8. "Recovery point"

The patent references "recovery points" several times in Claim 15, though not at all in Claim 1. Claim 15 describes a recomputing method; among its functions is "designating a set of recovery points in the optimized binary translated code during optimized translation of the foreign code, wherein each recovery point represents a foreign state." Patent at 17:25–23. Recovery points are also discussed in conjunction with documentation: "each documentation in the set of documentations corresponds to a recovery point in the optimized binary translated code and describes operations required to calculate a corresponding foreign state for the recovery point." *Id.* at 17:30–34. The claim then goes into detail about how a documentation and a recovery point work together: a documentation "correspond[s] to executed optimized binary translated code when an exception arises during its execution to recover a foreign state corresponding to a recovery point for the exception, wherein the foreign state is recovered by executing the operations for the one of the documentations." *Id.* at 18:2–7.

Here again, the defendants argue that their definition of the term is expressly laid out in the patent itself. They point to a passage of the specification stating that "a set of

'Recovery Points' (RP) are [sic] included in the binary translated code, with the following properties." Patent at 9:30–32. This passage then lists three properties, each of which defendants truncate and include in their definition of this term: "a point in the binary translated code that (1) has a correspondence with an instruction in the foreign code; (2) is described by a documentation and (3) any synchronous exception between adjacent recovery points can be reinvoked by interpreting foreign instructions starting from the previous recovery point." Defs.' Br. at 20. Defendants then argue that each of these properties is also reflected in Claim 15.

Cascades says the meaning of this term would be apparent to one of ordinary skill in this field. However, Cascades alternatively defines "recovery" with a dictionary definition: "restoration or return to a former[,] usual[,] or correct state or condition." Resp. at 22 (citing Pl.'s Ex. H (commas restored; Cascades omitted them)). Cascades also argues that, if plain meaning is not acceptable, "restoration point" is an appropriate definition "because a recovery point seeks to bring the host program configuration back to a correct dedicated foreign state in the case of an exception." *Id.* Cascades quotes two passages from the specification, one stating which type of information is saved at recovery points, and another defining what a "recovery mechanism" does. *See* Patent at 4:60–65, 17:16–19. From these passages, Cascades concludes that "[a] recovery point is thus a point at which saved information can be restored to assist the recovery mechanism." Resp. at 22. Further, Cascades says the defendants' definition "misstates the claims," citing one example: though defendants' definition states that "[e]very Recovery Point is described by a documentation set," Claim 15 says "each documentation in the set of documentations *corresponds to* a recovery point." Resp. at

22 (quoting Patent at 17:16–19 (emphasis added by Cascades)). At the claim construction hearing, Cascades emphasized that the defendants' construction is "not a definition," because it lists "features of a recovery point" but does not define it. Hrng. Trans. at 111.

Defendants reply that the Cascades definition is a "mere substitution of the word 'restoration' for 'recovery,'" and that only defendants' definition relies on actual language from the patent. Defs.' Repl. at 15. At the hearing, they observed that the patent refers to "Recovery Points" and that they draw their definition from a passage of the patent that similarly capitalizes the first letter in "Recovery" and in "Point." Defendants contended that the patentees were "consistent in referring back to the recovery point" in the way in which it is described in the defendants' definition. Hrng. Trans. at 110–111.

As with some other terms analyzed above, the difficulty with the defendants' definition is that none of the three features the definition ascribes to a recovery point actually states what a recovery point *is*. Parts one and two of the definition do not describe the nature of a recovery point, but rather say that a recovery point "has a correspondence with an instruction in the foreign code" and "is described by a documentation." No knowledge is gained of the actual nature of a recovery point from these descriptions. Part three is even less of a definition: "any synchronous exception between adjacent recovery points can be reinvoked by interpreting foreign instructions starting from the previous recovery point." Cascades, on the other hand, accomplishes little with its own definition ("restoration point"), in that it merely swaps out one word for another with little change in meaning.

The term "recovery point" is used with equal applicability in multiple situations in the patent. For example, in the "Summary of the Invention" section, the patent states that "state information is saved at a plurality of recovery points in the binary translated code" during the translation process. Patent at 4:60–61. At another point, the specification states that "[w]ith the information provided by documentation 528, exception handler 532 is able to recreate the state of the host computer system at the most recently executed recovery point." *Id.* at 15:25–28. Not all of these mentions of recovery points put the term in capital letters, contradicting the defendants' argument that the patentees intended an official, exclusive definition for the term. In fact, they point to a general, identifiable concept that is amenable to a simpler construction.

Although Cascades' proposed definition of "restoration point" is not helpful, Cascades goes further in its brief toward actually defining what a recovery point *is*. There, it included two passages from the patent discussing recovery points to conclude that a "recovery point is thus a point at which saved information can be restored to assist the recovery mechanism." Pl.'s Resp. at 22. Likewise, during their portion of the tutorial at the hearing, the defendants said that "a recovery point is set in the patent every time we're running an instruction that might cause an exception. So every time we're concerned we might be doing something that might go wrong, we set that as a recovery point." Hrng. Trans. at 35–36. (Elsewhere, the defendants described a recovery point as "the last known point where everything was okay." *Id.* at 36.) These statements are much clearer about the nature of a recovery point and the function it serves in the greater patent scheme. The Court therefore construes the term "recovery

point" to mean a "location in binary translated code that is marked prior to the occurrence of an exception."

Conclusion

The Court rules on the disputed claim terms as set forth in the foregoing decision. These cases are set for a further status hearing before the undersigned judge on January 21, 2014 at 9:30 a.m. The parties are directed to confer prior to that date regarding whether there are additional common issues that might warrant expansion of the IOP 13 pretrial consolidation and are to submit a status report containing their positions in this regard by no later than January 17, 2014. The status report should include each party's positions on what, if any, further fact discovery is appropriate under Local Patent Rule 1.3 as well as each party's intentions with regard to the disclosure of expert witnesses.

MATTHEW F. KENNELLY
United States District Judge

Date: January 2, 2013